WHAT IS CLAIMED IS:

1. An apparatus for creating a graphical user interface to allow user requirements for a computer program to be written by an automated software production tool to be entered and converted to a formal language specification, comprising:

a software-generating computer programmed to:

display a plurality of dialog boxes and/or graphic screens each of which has boxes which can be filled in with data or menu selections, tools or icons which can be invoked to allow a user to enter information defining classes, attributes, events, relationships between classes, valuation formulas for events that affect the value of variable attributes and all the other information needed to define a conceptual model of the requirements a computer program to be written by said software generation tool must comply with.

2. A method for using a computer to display a graphical user interface to allow user requirements for a computer program to be written by an automated software production tool to be entered, comprising:

displaying a plurality of dialog boxes and/or graphic screens each of which has boxes which can be filled in with data or menu selections, tools or icons which can be invoked to allow a user to enter information and/or create graphic objects which define classes, attributes, events, relationships between classes, valuation formulas for events that affect the value of variable attributes and all the other information needed to define a conceptual model of the requirements a computer program to be written by said software generation tool must comply with; and

as a user fills in data or makes selections or creates graphic objects, displaying the data filled in or selected and the graphic object created in the location on the dialog box and/or graphic screen where the data was filled in or selected or the graphic object was created.

3. The process of claim 2 further comprising the step of using a computer to automatically translate the data filled in or selected and/or graphic objects created into a specification for the computer program to be generated written in a formal language or other symbology which has predefined rules of syntax and semantics which can be used to verify that the specification so written is syntactically and semantically correct, complete and not ambiguous

4. A computer-readable medium containing instructions for controlling a computer system to display a graphical user interface through which a user can enter data to create a formal language specification defining a computer program, said specification to be automatically translated by a computer into a computer program that implements the requirements of said specification by:

displaying a plurality of dialog boxes and/or graphics screens and displaying boxes where data can be filled in, boxes where data can be chosen from a menu of choices, tools, icons or menu choices or some combination of the above in connection with display of said dialog boxes and/or graphics screens which allow a user to enter and/or select data and/or draw graphic objects to define classes of objects having attributes of fixed, variable and other types, and having services, and define mathematical and/or logical formulas controlling how services affect the values of variable attributes, and define relationships between classes, and enter data or draw graphics which represent all concepts necessary to complete a conceptual model of said computer program to be written.

5. The computer-readable medium of claim 4 further containing instructions for controlling a computer to automatically translate said specification into working computer code, by:

controlling said computer to automatically translating said conceptual model into a specification of said computer program written in a formal language or symbology having predefined rules of syntax and semantics;

controlling said computer to use said rules of syntax and semantics to validate said specification to verify that is syntactically and semantically complete, correct and not ambiguous; and

controlling said computer to translate said specification into working computer code.

6. A process carried out in a computer for translating a formal language specification stored in said computer's memory and defining the requirements for a user interface of a computer program, into working computer code that can control a computer to implement said user interface, comprising:

write code to display requests for a user name and password and receive inputs in response thereto and authenticate the user;

write code to determine the privilege level of a user who has logged in and determine the classes of objects, attributes and services this user has privileges to access, retrieve the appropriate data from said specification and display the appropriate system view to said user;

write code to link each service of each object to an appropriate object server program which can control a computer to carry out said service;

write code to display query/selection search forms to allow users to enter data to define a search for data instances that satisfy the search criteria entered by the user and conduct such a search when requested for all instances that satisfy the user-specified search criteria;

write code to determine automatically which services of an object can be invoked given the current state of the object and only allow those services to be invoked;

write code to furnish initial values for object-valued arguments of services and receive any user input arguments;

write code to check data type entered by a user for validity for the argument the data fills and make sure the entered data is within a valid range for the argument the data is intended to fill;

write code to check for dependencies between arguments, and, if a dependency exists, and user input data triggers the dependency, to enable/disable the dependent arguments or fill in values of the dependent arguments, and consequently triggering other dependency rules;

write code to invoke the appropriate object server code linked to a particular service when a user makes an input indicating a desire to invoke that service and to pass the object server code the appropriate arguments for the service;

write code to wait for results of execution of a service, and to display an error message if an error occurred, but, if no error occurred, to wait for further user input.

7. An apparatus for translating a formal language specification stored in said computer's memory and defining the requirements for a user interface of a computer program, into working computer code that can control a computer to implement said user interface, comprising:

a computer programmed to perform the following functions:

write code to display requests for a user name and password and receive inputs in response thereto and authenticate the user;

the appropriate data from said specification and display the appropriate system view to said user;

write code to link each service of each object to an appropriate object server program which can control a computer to carry out said service;

write code to display query/selection search forms to allow users to enter data to define a search for data instances that satisfy the search criteria entered by the user and conduct such a search when requested for all instances that satisfy the user-specified search criteria;

write code to determine automatically which services of an object can be invoked given the current state of the object and only allow those services to be invoked;

write code to furnish initial values for object-valued arguments of services and receive any user input arguments;

write code to check data type entered by a user for validity for the argument the data fills and make sure the entered data is within a valid range for the argument the data is intended to fill;

write code to check for dependencies between arguments, and, if a dependency exists, and user input data triggers the dependency, to display an appropriate dialog box prompting the user to enter input data needed to satisfy the dependency;

write code to invoke the appropriate object server code linked to a particular service when a user makes an input indicating a desire to invoke that service and to pass the object server code the appropriate arguments for the service;

write code to wait for results of execution of a service, and to display an error message if an error occurred, but, if no error occurred, to wait for further user input.

8. A computer-readable medium containing instructions to control a computer to translate a specification for a user interface for a computer program written in a formal language into computer code which can control a computer to implement the specified interface, by:

writing code to display requests for a user name and password and receive inputs in response thereto and authenticate the user;

writing code to determine the privilege level of a user who has logged in and determine the classes of objects, attributes and services this user has privileges to access, retrieve the appropriate data from said specification and display the appropriate system view to said user;

writing code to link each service of each object to an appropriate object server program which can control a computer to carry out said service;

writing code to display query/selection search forms to allow users to enter data to define a search for data instances that satisfy the search criteria entered by the user and conduct such a search when requested for all instances that satisfy the user-specified search criteria;

writing code to determine automatically which services of an object can be invoked given the current state of the object and only allow those services to be invoked;

writing code to furnish initial values for object-valued arguments of services and receive any user input arguments;

writing code to check data type entered by a user for validity for the argument the data fills and make sure the entered data is within a valid range for the argument the data is intended to fill;

writing code to check for dependencies between arguments, and, if a dependency exists, and user input data triggers the dependency, to display an appropriate dialog box prompting the user to enter input data needed to satisfy the dependency;

writing code to invoke the appropriate object server code linked to a particular service when a user makes an input indicating a desire to invoke that service and to pass the object server code the appropriate arguments for the service;

writing code to wait for results of execution of a service, and to display an error message if an error occurred, but, if no error occurred, to wait for further user input.